Software User Manual

BCM963XX

BCA CPE Software Board Parameters

963XX-SWUM402-R

5300 California Avenue • Irvine, CA 92617 • Phone: 949-926-5000 • Fax: 949-926-5203

July 28, 2014

Revision History

Revision	*Date*	*Change Description*
963XX-SWUM402-R	07/28/14	**Updated:** • “How Board Parameters Work” on page 10 • “Parameter References” on page 17 • “bp_ulGpioOverlay” on page 26 • Table 1: “BP_OVERLAY_SERIAL_LEDS GPIO Pin Assignments,” on page 27 • Table 2: “BP_OVERLAY_EPHY_LED_x GPIO Pin Assignments,” on page 27 • Table 3: “BP_OVERLAY_GPHY_LED_x GPIO Pin Assignments,” on page 27 • Table 4: “BP_OVERLAY_USB_LED GPIO Pin Assignments,” on page 28 • “bp_usGpioUart2Sdin” on page 30 • “bp_ulInterfaceEnable” on page 53 **Added:** • Table 10: “BCM6328 Switch Configuration,” on page 34 • “bp_usPhyConnType” on page 38 • “bp_usPhyDevName” on page 39 • “bp_ulPortMaxRate” on page 39 • “bp_usGpio_Intr” on page 44 • “bp_usButtonIdx” on page 45 • “bp_usButtonExtIntr” on page 46 • “bp_usButtonAction” on page 46 • “bp_usCfeResetToDefaultBtnIdx” on page 47 • “bp_usGpioSpiSlaveReset” on page 49 • “bp_usSpiSlaveSelectGpioNum” on page 50 • “bp_usSgmiiDetect” on page 53 **Removed:** • bp_usGpioLaserDis • bp_usGpioLaserTxPwrEn • Table 11: “BCM6368 Switch Configuration” • Table 13: “BCM6818G Switch Configuration”

Revision	*Date*	*Change Description*
963XX-SWUM401-R	18/12/13	**Updated:** • Scope of “bp_ulGpioOverlay” on page 25 • “bp_usGpioUart2Sdin” on page 31 • “Adding Support for a New Board” on page 11 **Added:** • “PinMux Check Utility” on page 14 • “Crossbar Switch Topology” on page 15 • New Parameters: “bp_usGpioLedReserved” on page 18 “bp_usSerialLedData/Clk/Mask” on page 29 “bp_usGphyBaseAddress” on page 38 “bp_ulInterfaceEnable” on page 50 • Table 14: “BCM63138 Switch Configuration,” on page 36 • BCM6838X chip family **Removed:** • BCM6828 Switch Configuration Table • BCM6816 Switch Configuration Table • bp_ulNonPeriphGpioCtrlMap
963XX-SWUM400-D1	03/16/12	Initial release

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

Table of Contents

List of Tables

About This Document

Purpose and Audience

The Broadcom BCA CPE reference software provides support for multiple hardware reference design boards with a single image.

This document describes the detailed usage of board parameters, provides instructions for supporting a new board, and lists board parameter references. It is intended for hardware and software engineers.

Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

For a comprehensive list of acronyms and other terms used in Broadcom documents, go to: http://www.broadcom.com/press/glossary.php.

Document Conventions

The following conventions may be used in this document:

Convention	*Description*
Bold	User input and actions: for example, type **exit**, click **OK**, press **Alt+C**
`Monospace`	Code: `#include <iostream>` HTML: `<td rowspan = 3>` Command line commands and parameters: `wl [-l] <command>`
< >	Placeholders for *required* elements: enter your <username> or `wl <command>`
[ ]	Indicates *optional* command-line parameters: `wl [-l]` Indicates bit and byte ranges (inclusive): [0:3] or [7:0]

References

The references in this section may be used in conjunction with this document.

Note: Broadcom provides customer access to technical documentation and software through its Customer Support Portal (CSP) and Downloads and Support site (see Technical Support).

For Broadcom documents, replace the “xx” in the document number with the largest number available in the repository to ensure that you have the most current version of the document.

Document (or Item) Name	Number	Source
Broadcom Items		
[1] *Using GPIO as LED Drivers and Other Alternate Functions*	6316X_6326X-AN1XX-R	Broadcom CSP
[2] *Using GPIO as LED and/or NAND Flash Drivers*	636X_6328X-AN1XX-R	Broadcom CSP
[3] *Configuring AFE_ID for xDSL PHY Firmware*	63XX-AN7XX-R	Broadcom CSP
[4] *Run-time Selection of Voice Board Parameters*	DSLxChange-AN1XX-R	Broadcom CSP
[5] *Connect and Configure the BCM6838X LEDs*	6838X-AN1XX-R	Broadcom CSP

Technical Support

Broadcom provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates through its customer support portal (https://support.broadcom.com). For a CSP account, contact your Sales or Engineering support representative.

In addition, Broadcom provides other product support through its Downloads and Support site (http://www.broadcom.com/support/).

Overview

The Broadcom BCA CPE reference software provides the support for multiple hardware reference design boards with a single image. The software image contains multiple board parameters that determine the board configuration—such as Ethernet switch, LED assignment, GPIO control, and GPON/EPON parameters—for each reference board. Customers need only add a new set of board parameters, with no function code change, in order to support a new board based on a mix of features in existing reference designs.

Working with Board Parameters

How Board Parameters Work

The board parameters, structure, and constants are defined in:

```
<source tree top level>\shared\opensource\boardparms\bcm963xx\boardparms.c
<source tree top level>\shared\opensource\boardparms\bcm963xx\bp_defs.h
<source tree top level>\shared\opensource\boardparms\bcm963xx\bp_funcs.c
<source tree top level>\shared\opensource\include\bcm963xx\boardparms.h
```

Each board parameter instance defines the board configurations for one particular reference board. It consists of a variable length of board parameter elements defined as:

```
typedef struct bp_elem {
  enum bp_id id;
  union {
    char * cp;
    unsigned char * ucp;
    unsigned char uc;
    unsigned short us;
    unsigned long ul;
  } u;
} bp_elem_t;

static bp_elem_t g_bcm963168xh[] = {
  {bp_cpBoardId,              .u.cp = "963168XH"},
  {bp_usGpioOverlay,          .u.ul =(BP_OVERLAY_SERIAL_LEDS |
                                      BP_OVERLAY_USB_DEVICE|
                                      BP_OVERLAY_PCIE_CLKREQ |
                                      BP_OVERLAY_HS_SPI_SSB5_EXT_CS)},
  {bp_usGpioLedAdsl,          .u.us = BP_GPIO_13_AH},
  {bp_usGpioLedSesWireless,   .u.us = BP_SERIAL_GPIO_7_AL},
  {bp_usGpioLedWanData,       .u.us = BP_GPIO_8_AL},
  {bp_usGpioLedWanError,      .u.us = BP_SERIAL_GPIO_2_AL},
  {bp_usGpioLedBlPowerOn,     .u.us = BP_GPIO_10_AL},
  {bp_usGpioLedBlStop,        .u.us = BP_SERIAL_GPIO_3_AL},
  {bp_usExtIntrResetToDefault, .u.us = BP_EXT_INTR_0},
  {bp_usExtIntrSesBtnWireless, .u.us = BP_EXT_INTR_1},
  {bp_usAntInUseWireless,     .u.us = BP_WLAN_ANT_MAIN},
  {bp_usWirelessFlags,        .u.us = 0},
```

```
    {bp_ucPhyType0,             .u.uc = BP_ENET_EXTERNAL_SWITCH},
    {bp_ucPhyAddress,           .u.uc = 0x0},
    {bp_usConfigType,           .u.us = BP_ENET_CONFIG_MMAP},
    {bp_ulPortMap,              .u.ul = 0x58},
    {bp_ulPhyId3,               .u.ul = BP_PHY_ID_4},
    {bp_ulPhyId4,               .u.ul = RGMII_DIRECT | EXTSW_CONNECTED},
    {bp_ulPhyId6,               .u.ul = BP_PHY_ID_25},
    {bp_ucPhyType1,             .u.uc = BP_ENET_EXTERNAL_SWITCH},
    {bp_ucPhyAddress,           .u.uc = 0x0},
    {bp_usConfigType,           .u.us = BP_ENET_CONFIG_HS_SPI_SSB_5},
    {bp_ulPortMap,              .u.ul = 0x0f},
    {bp_ulPhyId0,               .u.ul = BP_PHY_ID_0},
    {bp_ulPhyId1,               .u.ul = BP_PHY_ID_1},
    {bp_ulPhyId2,               .u.ul = BP_PHY_ID_2},
    {bp_ulPhyId3,               .u.ul = BP_PHY_ID_3},
    {bp_ucDspType0,             .u.uc = BP_VOIP_MIPS},
    {bp_ucDspAddress,           .u.uc = 0},
    {bp_usGpioVoip1Led,         .u.us = BP_SERIAL_GPIO_4_AL},
    {bp_usGpioVoip2Led,         .u.us = BP_SERIAL_GPIO_5_AL},
    {bp_usGpioPotsLed,          .u.us = BP_SERIAL_GPIO_6_AL},
   {bp_ulAfeId0,            .u.ul = BP_AFE_CHIP_6306 | BP_AFE_LD_ISIL1556 | BP_AFE_FE_AVMODE_VDSL
 | BP_AFE_FE_REV_12_21 | BP_AFE_FE_ANNEXA },
    {bp_usGpioExtAFEReset,      .u.us = BP_GPIO_11_AL},
    {bp_last}
};
```

The BCM963168XH example shown above defines the board parameters for the BCM963168XH reference board. It contains the board parameter elements identified by the board parameter ID, such as board ID (*bp_cpBoardId*) and GPIO overlay (*bp_usGpioOverlay*), that are necessary to define the hardware configuration for the software to run properly.

Multiple board parameter structures are defined for every chip family, providing a distinct set of parameters for each different reference design board used with that chip family. They are grouped into a new array:

```
static bp_elem_t * g_BoardParms[] =
{g_bcm96368vvw, g_bcm96368mvwg, … };
```

Each chip family defines its own *g_BoardParms* array. A compile flag controls which board parameter is compiled into which target profile image.

Both the CFE and Linux® images contain a copy of the board parameters and they share the same source of boardparms.c. When changing the board parameters make sure that both the CFE and Linux images are recompiled, to ensure the changes take effect in both places.

When the CFE and Linux kernels start:

1. They read the board ID string *NvramData.szBoardId* from the NVRAM and call the *BpSetBoardId*(*NvramData.szBoardId*) function.
   This function matches the input string to the *bp_cpBoardId* from each entry in the board parameters array *g_BoardParms* and sets the *g_pCurrentBp* to the matching board parameter entry.
2. *g_pCurrentBp* is then pointed to the correct board parameter set and used for all the board parameters accessed from CFE and Linux kernel.

Users can use the **b** command under the CFE prompt to change the board ID string to run the same image on a different reference board.

The boardparms.c module also provides all the helper functions for retrieving the board parameter ID properties. Each board parameter ID has a corresponding helper function. For example, *BpGetGPIOverlays* reads the GPIO overlay (*bp_usGpioOverlay)* property. The CFE and Linux kernels use these helper functions to get the LED, GPIO assignment, and Ethernet switch/PHY connection information etc, so that they can initialize and use the board correctly.

Adding Support for a New Board

The board's design architecture makes the process of adding a new board an easy task. The best practice approach is to copy the board parameters from the reference board that most closely matches the new design and modify them. The example below shows the steps to add support for a new board. It uses the BCM963168XH as the reference board and adds a new board called BCM963168EXP.

3. Copy the BCM963168XH board parameter **g_bcm963168xh** and make a new entry **g_bcm963168exp** in the BCM63268 section identified by the _BCM963268_ compiling flag.
4. Change the first board parameter ID, **bp_cpBoardId**, to the new board name.

```
{bp_cpBoardId,                .u.cp = "963168EXP"},
```

5. Add **g_bcm963168exp** to the *g_BoardParms* array in the same section.

```
static bp_elem_t * g_BoardParms[] = {g_bcm963168xh, g_bcm963168exp,… };
```

6. If the internal WLAN is used, make a new entry of SROM PATCH by copying the SROM patch from a board with similar RF characteristics in the *wlanPaInfo* array. Change the board ID to **963168EXP** in the new entry. Contact Broadcom Support if the change to the SROM parameter value is needed.

Note: Care should be taken when using an external power amplifier. It is important to choose an appropriate SROM patch from devices with an external power AMP; otherwise the device will send its full output power into the input of the external PA.

7. If wireless is used in the new design:
   a. Make a new entry for the wireless LAN PCI ID by copying the BCM963168XH entry in the *wlanPciInfo* array.
   b. In the new entry, change the board ID to **963168EXP**.
   c. If needed, change the wireless PCI ID and sub ID.
8. Check for changes to the LED assignments, such as GPIO pin number changes, a GPIO pin output changing to serial LED output, or any added or removed LEDs. See "LED Changes" on page 14.
9. Check for any GPIO assignment changes, such as GPIO pin number changes and any added or removed GPIO pins. In this example, the BCM963168XH board uses GPIO 11 as an external AFE reset. If this is changed to another pin, the pin assignment must be updated accordingly.

```
{bp_usGpioExtAFEReset,        .u.us = BP_GPIO_11_AL}
```

- Check for a change to the GPIO override function. See "GPIO Override" on page 15.
- Check for changes to the Ethernet switch and PHY connection topology.

  Review the new design and make the respective change for these switch parameters. See "Ethernet Switch and PHY Topology" on page 15 for the detailed explanation of switch parameters and their usage.
- Check the AFE connections for DSL chips.
  BCM63268 can support two VDLS2 interfaces. One uses the internal AFE and the other uses an external AFE.
  Based on the hardware design, the user must specify *bp_ulAfeId0* and *bp_ulAfeId1* if a second AFE interface is used. Each interface properties must be defined, and which one is internal and which is external. Refer to *Configuring AFE_ID for xDSL PHY Firmware* Reference [3] on page 8 for details about this parameter for the BCM63XX chips.
- Check the GPON/EPON settings for GPON/EPON chips. For GPON or EPON, there are other parameters to check: *bp_usGponOpticsType* and *bp_cpDefaultOpticalParams*. In certain chips, the user must specify the PHY ID to indicate if it is the GPON/EPON WAN interface. See the description of bp_ulPhyIdn for details of this requirement.
- Check external interrupt pin usage.

  Example: The BCM963168XH uses two external interrupt inputs.

  ```
  {bp_usExtIntrResetToDefault, .u.us = BP_EXT_INTR_0},
  {bp_usExtIntrSesBtnWireless, .u.us = BP_EXT_INTR_1},
  ```

  If the new design changes the usage of external interrupt, these two parameters need to be updated.

  Example: In the BCM6838X chips each external interrupt input can be driven from any GPIO. The board BCM968380FHGU uses two external interrupt inputs.

  ```
  {bp_usExtIntrResetToDefault, .u.us = BP_EXT_INTR_0},
  {bp_usGpio_Intr,             .u.us = BP_GPIO_72_AL},
  {bp_usExtIntrSesBtnWireless, .u.us = BP_EXT_INTR_1},
  {bp_usGpio_Intr,             .u.us = BP_GPIO_47_AL},
  ```

  In this example, external interrupt 0 is driven from GPIO 72, and external interrupt 1 is driven from GPIO 47.
- If a voice daughtercard is used, check the voice board parameters. Refer to the *Run-time Selection of Voice Board Parameters* (Reference [4] on page 8), for details.
- Build the CFE and a system image to test on the new board.

LED Changes

In the BCM63268, the LED controller takes two input sources: hardware driven LED and software controlled LED. There are a total of 24 LED bits in the LED controller. The hardware driven LEDs use fixed mappings. For example, GPHY SPEED LED 0 is always mapped to LED 0. Software has the flexibility to assign the rest of the LEDs to any functions. The LED controller drives the LED diode through a GPIO pin or a serial shift register and then the shift register drives the actual diode LED. When driven to the GPIO pin, LED bits 0 to 23 maps to GPIO pins 0 to 23 respectively. When driving a serial shift register, only the first 16 LED bit are captured and shifted to serial register output 0 to 15 respectively.

There are certain constraints on the LED and GPIO usage and it differs between different chip families:

- For BCM6316X/BCM6326X, refer to *Using GPIO as LED Drivers and Other Alternate Functions* (Reference [1] on page 8)
- For BCM636X/BCM6328X refer to *Using GPIO as LED and/or NAND Flash Drivers* (Reference [2] on page 8) for details regarding LED/GPIO usage, mapping, constraints, and limitations.
- For BCM6838X refer to *Connect and Configure the BCM6838X LEDs* (Reference [5] on page 8)

In our example, if the new design changes the WAN error LED to use GPIO 9 directly output, then we need to change the *bp_usGpioLedWanError* value:

```
{bp_usGpioLedWanError,       .u.us = BP_GPIO_9_AL},
```

However, if a new design uses EPHY1 from the chip's internal switch, LED bit 9 is hard wired to the EPHY1 LINK/ACT LED function and GPIO 9 cannot be used for WAN errors. This is true even when a shift register is used for the EPHY1 LINK/ACT LED because direct GPIO and shift register outputs share the same source LED register bit (bit 9).

If any new LEDs are added in the new design, a new element must be defined in the board parameters defining the new LED assignment. For example, when secondary DSL is used, the following code must be added in order to map the second DSL line LED with the proper BP_GPIO or BP_SERIAL_GPIO assignment based on the hardware:

```
{bp_usGpioSecLedAdsl,        .u.us = BP_XXX}
```

The new entry should be appended after the first DSL line LED ID, *bp_usGpioLedAdsl* (even though it can be added anywhere in the *g_bcm963168exp* array, it is not recommended to do so.)

If no second FXS station used, simply remove the following parameter ID:

```
{bp_usGpioVoip2Led,          .u.us = BP_SERIAL_GPIO_5_AL}
```

For the complete LED board parameter ID list and usage, see "LED Parameter ID Definitions" on page 19.

GPIO Override

Besides the usage of the regular GPIO control and LED control, the GPIO pin can be overridden for other functions. The *bp_usGpioOverlay* parameter or the *bp_usSerialLedData/Clk/Mask* functions tell the software what other functions are used on the GPIO pin. For example, the BCM963168XH use GPIO pins 0 and 1 as the serial LED output.

If the board does not use a serial shifted LED output, *BP_OVERLAY_SERIAL_LEDs* should be removed from the overlay bitmap. Each overlay function uses fixed GPIO pin assignments dictated by the chip hardware. The user does not need to specify which GPIO pin is used for that function.

For the list of overlay functions and a detailed description of their usage, see “bp_ulGpioOverlay” on page 26. For the BCM63138, BCM63148, BCM63381 chips, see “bp_usSerialLedData/Clk/Mask” on page 29 that replaces the bp_ulGpioOverlay function.

This function is not relevant for not relevant for BCM6838.

PinMux Check Utility

The BCM63138, BCM63148, and BCM63381 have the PinMux test utility, *bptest*, which is built and run during the build process. The utility checks for the most common errors in boardparms, such as requesting that a hardware-controlled LED be configured to a pin other than one of those that the hardware is capable of managing.

If running bptest causes a build to abort, carefully check your boardparms settings against the chip data sheet.

Ethernet Switch and PHY Topology

The Broadcom broadband access chips have an internal switch with integrated transceiver (PHY) and external transceiver. They also support an external switch cascaded behind the internal switch. A maximum of two Ethernet switches (one internal and one external) are supported per board. On some chips, like the BCM6362 and BCM6368, the internal switch port cannot be used when an external switch is connected.

On a BCM62368, a maximum of eight ports are supported in the internal switch with three 10/100 Fast Ethernet internal PHY (ports 0–2, PHY ID 1–3), one internal Gigabit PHY (port 3, PHY ID 4), and four RGMII/MII ports (ports 4–7, external PHY ID) for connection with an external PHY or switch.

Port configuration, mapping, and PHY ID information is required for the board parameter design. Refer to the chipset or external switch data sheet for device-specific details. Table 9 through Table 11 list the internal switch configurations for all the supported chips.

The BCM963168XH board supports a BCM53125 external switch. Each switch takes a switch and PHY parameter block, which consists of the following parameter IDs:

- *bp_ucPhyType0*/1
- *bp_ucPhyAddress*
- *bp_usConfigType*
- *bp_ulPortMap*
- *bp_ulPhyId*

The internal switch uses *bp_ucPhyType0* and the external switch uses *bp_ucPhyType1*. The *bp_ucPhyType0* and *bp_ucPhyType1* parameters are always set to BP_ENET_EXTERNAL_SWITCH and *bp_ucPhyAddress* is set to zero. *bp_usConfigType* is set to BP_ENET_CONFIG_MMAP for the internal switch. For the external switch, *bp_usConfigType* can be BP_ENET_CONFIG_HS_SPI_SSB_x, BP_ENET_CONFIG_SPI_SSB_x or BP_ENET_CONFIG_MDIO, depending on the management mechanism for the external switch.

The *bp_ulPortMap* parameter defines the bitmap of the ports used in the switch. BCM963168XH uses GPHY1 (port 3), RGMII 1 (port 4), and RGMII 3 (port 6) in the internal switch, so the mapping is 0x58. For each port that is used, *bp_ulPhyIdx* must be specified. So *bp_ulPhyId3*, *bp_ulPhyId4*, and *bp_ulPhyId6* are set.

Port 3 uses the internal PHY, so it has the internal PHY ID.

Port 4 is connected to an external switch so no PHY ID is needed, but *bp_ulPhyId4* must be set with RGMII_DIRECT | EXTSW_CONNECTED flags so that the software knows that it is connected to an external switch and uses the RGMII interface.

Port 6 is connected to an external PHY, based on the hardware design. The external PHY ID is 25.

Crossbar Switch Topology

The BCM63138 and BCM63148 chips have a crossbar switch connected to port 0 of the Runner network processor and port 4 of the switch core. PHY configuration corresponds to each port of the crossbar switch rather than to the port to which the switch is connected internally.

On crossbar-enabled ports, the *bp_ulPhyIdX* corresponding to the port does not specify the PHY ID but, marks the beginning of one or more Crossbar port descriptions. Each of these port descriptions list their own PHY ID and related parameters.

```
{bp_ulPhyId0, .u.ul = BP_PHY_ID_NOT_SPECIFIED}, // This has one crossbar setting
   {bp_ulCrossbar, .u.ul = 10}, // This is crossbar port 10
       {bp_ulCrossbarPhyId, .u.ul = (BCM963138_PHY_BASE + 0x04) }, // same values as bp_ulPhyIdX
       {bp_usSpeedLed100, .u.us = BP_GPIO_26_AL},
       {bp_usSpeedLed1000, .u.us = BP_GPIO_27_AL},
       {bp_usLinkLed, .u.us = BP_GPIO_14_AL},
{bp_ulPhyId1, .u.ul = GMII_DIRECT | EXTSW_CONNECTED}, // here the next port begins
```

Alternatively:

```
{bp_ulPhyId0, .u.ul = BP_PHY_ID_NOT_SPECIFIED}, // This has more than one crossbar setting
   {bp_ulCrossbar, .u.ul = 10},
       {bp_ulCrossbarPhyId, .u.ul = (BCM963138_PHY_BASE + 0x04) },
       {bp_usSpeedLed100, .u.us = BP_GPIO_26_AL},
       {bp_usSpeedLed1000, .u.us = BP_GPIO_27_AL},
   {bp_ulCrossbar, .u.ul = 11},
       {bp_ulCrossbarPhyId, .u.ul = 0x1 | PHY_INTEGRATED_VALID | MAC_IF_RGMII_1P8V | PHY_EXTERNAL},
       {bp_usLinkLed, .u.us = BP_GPIO_14_AL},
{bp_ulPhyId1, .u.ul = GMII_DIRECT | EXTSW_CONNECTED}, // here the next port begins
```

Parameter References

This section lists all the supporting parameters and explains their usage. All chips listed in the chip families below are included. Each of these devices represents the primary chip ID of a family. The BCM63268 chip ID, for example, includes the BCM63168, BCM63169, BCM63268, and BCM63269 devices.

Chip Family

BCM6328
BCM6362
BCM63268
BCM6838
BCM63138
BCM63381

General Parameter ID Definitions

bp_cpBoardId

Scope

All chips

Type

String

Description

bp_cpBoardId defines the board ID string. The maximum string length is 16.

This parameter must appear first on the list.

bp_cpComment

Scope

All chips

Type

String

Description

bp_cpComment defines additional text information for this board.

bp_elemTemplate

Scope

All chips

Type

pointer to *bp_elem_t*

Description

bp_elemTemplate allows the user to include the parameters from another board. This reduces the size of the code and removes redundant definitions from boardparam.c. The example below defines the board parameters for BCM963168VX_P300, which inherits all parameters from the board BCM963168VX with the exception of *bp_cpBoardId* and *bp_ulAfeId0*, which are different in the new board.

```
static bp_elem_t g_bcm963168vx_p300[] = {
  {bp_cpBoardId,               .u.cp = "963168VX_P300"},
  {bp_ulAfeId0,                .u.ul = BP_AFE_CHIP_INT | BP_AFE_LD_6302 | BP_AFE_FE_ANNEXA |
BP_AFE_FE_REV_6302_REV_7_2_30},
  {bp_elemTemplate,            .u.bp_elemp = g_bcm963168vx},
  {bp_last}
};
```

bp_last

Scope

All chips

Type

None

Description

This parameter ID must be the last ID in a board parameter set.

LED Parameter ID Definitions

An LED can be assigned directly to a GPIO pin or external shift register using the serial LED driver function depending on the hardware design. Certain hardware-controlled LEDs are hard wired to fixed GPIO pins. For example, the WAN activity LED uses GPIO 8 in the BCM63268. These hard wired GPIO pins cannot be used for other LED function. The WLAN and USB device LEDs are also controlled by hardware directly and use other dedicated pins, so they don't need to specified with LED parameter ID.

Most of the GPIO pins are multiplexed with alternate functions and if an alternate function is enabled they can't be used to drive an LED. Refer to *Using GPIO as LED Drivers and Other Alternate Functions* (Reference [1] on page 8) for BCM6316X/BCM6326X and *Using GPIO as LED and/or NAND Flash Drivers* (Reference [2] on page 8) for BCM636X/BCM6328X for more information.

The value of the LED parameter ID can be one of the following:

- BP_SERIAL_GPIO_*n*_AL: LED connects to serial shift register and is active low.
- BP_SERIAL_GPIO_*n*_AH: LED connects to serial shift register and is active high.
- BP_GPIO_*n*_AL: LED connects to GPIO pin and is active low.
- BP_GPIO_*n*_AH: LED connects to GPIO pin and is active high.

where *n* is the GPIO pin number. The maximum value is device-dependent; check the specific device data sheet for details.

If the GPIO pin that drives the LED is also boot strap pin, the user should always use BP_GPIO_*n*_AL or BP_SERIAL_GPIO_*n*_AL because of the auto-inversion function built into hardware.

The internal Ethernet PHY link, activity, and speed LEDs are controlled by the GPIO Overlay in "GPIO Parameter ID Definitions" on page 26 and/or Ethernet switch parameter IDs in section "Ethernet Switch Parameter ID Definitions" on page 32.

bp_usGpioLedReserved

Scope

BCM63138, BCM63148, BCM63381

Type

Unsigned short

Description

Indicates the assignment of an LED function (serial or parallel) without telling the software to do anything with it. This will cause hardware initialization to configure the bit as an LED and initially turn it off. Other software can then control the function by writing to the appropriate bit of the LED register.

bp_usGpioLedAdsl

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned short

Description

bp_usGpioLedAdsl defines the output pin assignment for ADSL line link up status LED.

bp_usGpioLedAdslFail

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for ADSL line link down status LED. If it is not defined, software blinks *bp_usGpioLedAdsl* LED to indicate link down.

bp_usGpioSecLedAdsl

Scope

BCM6368, BCM63628

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for second ADSL line link up status LED.

bp_usGpioSecLedAdslFail

Scope

BCM6368, BCM63628

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for second ADSL line link down status LED. If it is not defined, software blinks *bp_usGpioSecLedAdsl* LED to indicate link down.

bp_usGpioLedSesWireless

Scope

Any board with wireless LAN support.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for the Wi-Fi Protected Setup LED.

bp_usGpioLedWanData

Scope

BCM6328, BCM6368, BCM6362, BCM63268, BCM6816, BCM6818

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for internet activity LED. On some chips this is a hardware-controlled LED:

- For the BCM6362 and BCM6328, it always uses GPIO 1 or serial LED bit 1.
- For the BCM6368, it always uses GPIO 5 or serial LED bit 5.
- For the BCM63268, it always uses GPIO 8 or serial LED bit 8.

On the above chips, this parameter can only be set to use the hard wired GPIO number.

For the BCM6818 and BCM6816, it is software-controlled LED, so users need to specify it with any available GPIO pin.

For the BCM6828, the EPON internet activity LED is controlled by EPON hardware module directly through one of the GPIO pins between GPIO 42 to GPIO 51. This pin is selected by the EPON firmware configuration. The hardware makes the connection to the LED but there is no need to specify it with this board parameter in software.

bp_usGpioLedWanError

Scope

BCM6328, BCM6368, BCM6362, BCM63268, BCM6816, BCM6818

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for internet connection error LED. If it is not defined, software blinks *bp_usGpioLedWanDataLED* to indicate a connection error. For the BCM6828, the EPON Internet error LED is controlled directly by the EPON hardware module through one of the GPIO pins between GPIO 42 to GPIO 51 This pin is selected by the EPON firmware configuration. The hardware makes the connection to LED but there is no need to specify it with this board parameter in software.

bp_usGpioLedBlPowerOn

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for Power ON LED.

bp_usGpioLedBlStop

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for POST Fail LED. This LED will also illuminate if the CFE enters its interactive console mode.

bp_usGpioLedGpon

Scope

BCM6816, BCM6818

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for GPON link up status LED.

bp_usGpioLedGponFail

Scope

BCM6816, BCM6818

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for GPON link down status LED. If it is not defined, software blinks *bp_usGpioLedGpon* to indicate GPON link down error.

The EPON link up and down status LEDs on a BCM6828 are controlled directly by the EPON hardware module through one of the GPIO pins between GPIO 42 to GPIO 51, selected by the EPON firmware configuration. Hardware makes the connection to the LED but there is no need to specify it in the board parameters.

bp_usGpioLedMoCA

Scope

BCM6829, BCM6816, BCM6819

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for MoCA connection up LED.

bp_usGpioLedMoCAFail

Scope

BCM6829, BCM6816, BCM6819

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for MoCA connection down LED. If it is not defined, software blinks *bp_usGpioLedMoCA* to indicate MoCA connection error.

bp_usGpioLedVoip

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID is not used.

bp_usGpioVoip1Led

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for the first VoIP status LED.

bp_usGpioVoip1LedFail

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for the first VoIP status failure LED. If it is not defined, software blinks *bp_usGpioVoip1Led* to indicate the first status failure.

bp_usGpioVoip2Led

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for the second VoIP status LED.

bp_usGpioVoip2LedFail

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for the second status failure LED. If it is not defined, software blinks *bp_usGpioVoip2Led* to indicate the second status failure.

bp_usGpioPotsLed

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for FXO POTS line status LED.

bp_usGpioDectLed

Scope

Any board that supports voice card.

Type

Unsigned short

Description

This parameter ID defines the output pin assignment DECT status LED.

GPIO Parameter ID Definitions

The value of the GPIO parameter IDs (except for *bp_usGpioOverlay or bp_usSerialLedData/Clk/Mask*) can be one of the following:

- BP_GPIO_*n*_AL, which assigns the function to use GPIO pin *n* and is active low.
- BP_GPIO_*n*_AH, which assigns the function to use GPIO pin *n* and is active high.

where *n* is the GPIO pin number and its maximum value is device dependent. Refer to the device-specific data sheet for more details.

bp_ulGpioOverlay

Scope

BCM6318, BCM6328, BCM6362, BCM63268

Type

Unsigned long

Description

This parameter ID defines the bitmap indicating what functions override the regular GPIO function in this board design. The GPIO pins that are overridden with other functions are chip dependent and are not specified in this parameter. For example, when BP_OVERLAY_SERIAL_LEDS bit is set in this parameter, GPIO pins 0 and 1 are overridden with serial LED clock and data pins in the BCM63268, but in the BCM6362 GPIO pins 2 and 3 are overridden. The Broadcom software handles this chip dependency transparently for the user, but the user must ensure that the hardware design connects the right GPIO pins to the shift register's data and clock input.

The value of this parameter is the logic OR of any of the following overlay options. The details of the overlay options are described below.

Overlay Options

BP_OVERLAY_PHY

This overlay option only applies to the BCM6368 and BCM63268 devices. It enables the GPIO pins for DSL PHY AFE mode and power control.

When this option is set in the BCM6368, it enables GPIO 0 as AFE mode selection and GPIO 1 as AFE power enable signal.

When this option is set in the BCM63268, it enables and selects the pair of GPIO 10 and GPIO 11 for internal AFE mode and power control, and the pair of GPIO 12 and GPIO 13 for external AFE mode and power control by default.

User can use this option with the AFE mode and power control parameters, described in "DSL AFE Parameter ID Definitions" on page 40, to manually select the GPIO pins.

BP_OVERLAY_SERIAL_LEDS

This overlay option applies to all chips. It enables two GPIO pins as serial LED clock and data pins. The GPIO pin number is chip dependent. Table 1 shows the pin assignment on each chip.

Table 1: BP_OVERLAY_SERIAL_LEDS GPIO Pin Assignments

Chip	Serial Clock	Serial Data
BCM6362	GPIO_2	GPIO_3
BCM6328	GPIO_7	GPIO_6
BCM63268	GPIO_0	GPIO_1

BP_OVERLAY_EPHY_LED_0

BP_OVERLAY_EPHY_LED_1

BP_OVERLAY_EPHY_LED_2

BP_OVERLAY_EPHY_LED_3

These overlay options apply to the BCM6362, BCM6328, and BCM63268.

The BCM63268 does not have a BP_OVERLAY_EPHY_LED_3 option.

These options override GPIO pins to drive internal EPHY link activity and speed LEDs. The GPIO pin number is chip dependent. Table 2 shows the GPIO pin assignments and corresponding LED register bits (if any) for the LED of EPHYs. See Table 9 through Table 11 for the mapping between these flags and switch port.

For the BCM63268, if a GPIO pin has a corresponding LED register bit, the user does not need to specify the related *BP_OVERLAY_EPHY_LED* x flag if they use the serial shift register to drive the LED diode. The corresponding GPIO pin can then be used for other non-LED purposes.

Table 2: BP_OVERLAY_EPHY_LED_x GPIO Pin Assignments

Chip	LINK ACT LED	SPEED LED
BCM6362	LED_0 to LED_3 Only	GPIO_4 to GPIO_7/LED_4 to LED_7
BCM6328	GPIO_25 to GPIO_28 Only	GPIO_17 to GPIO_20/LED_17 to LED_20
BCM63268	GPIO_9 to GPIO_11/LED_9 to LED_11	GPIO_13 to GPIO_15/LED_13 to LED_15

BP_OVERLAY_GPHY_LED_0

BP_OVERLAY_GPHY_LED_1

These overlay options apply to BCM63268. And BCM63268 does not have BP_OVERLAY_GPHY_LED_1 option. These options override GPIO pins to drive internal GPHY link activity and speed LEDs. For BCM6828F with EPHY only, these flags actually apply to the EPHY on switch port 2 and port 3. The GPIO pin number is chip dependent. Table 1 shows details of GPIO pin assignments and corresponding LED register bits (if any) for the LED of GPHY 0 to GPHY 1. See Table 9 through Table 11 for the mapping between these flags and switch port.

Table 3: BP_OVERLAY_GPHY_LED_x GPIO Pin Assignments

Chip	GPHY 0 LINK ACT	GPHY 1 LINK ACT	GPHY 0 SPEED (SPD0/SPD1)	GPHY 1 SPEED (SPD0/SPD1)
BCM63268	GPIO_12/LED_12	N/A	GPIO_0, 1/LED_0, 1	N/A

BP_OVERLAY_INET_LED

This overlay option is required for the BCM6368. It is optional for other chip. The bp_usGpioLedWanData parameter actually enables the GPIO function override.

BP_OVERLAY_MOCA_LED

This overlay options applies to BCM6816. It overrides one GPIO pin as the MoCA LED. The GPIO pin number is chip dependent. Refer to its data sheet for detail of pin assignment.

BP_OVERLAY_USB_LED

This overlay option applies to BCM6362, BCM6328, and BCM63268. It overrides one GPIO pin as the USB device LED function in some chips. Other chips use hardware dedicate pin. See table below for the detail of the pin assignment. There is no USB host LED. This option should use with BP_OVERLAY_USB_DEVICE. Table 4 shows the GPIO pin assignments.

Table 4: BP_OVERLAY_USB_LED GPIO Pin Assignments

Chip	*Device Pin*	*LED Bit*	*Notes*
BCM6362	GPIO_0	LED 0	Routed through LED 0. Overlay flag required flag required
BCM6328	USB_DEVICE_LED/ USB_PWRON	N/A	Hard wired. Overlay flag NOT required
BCM63268	USB_PWRON2/ DEVICE_LED	LED 23	Hard wired but routed through LED 23. Overlay flag required

BP_OVERLAY_USB_DEVICE

All chips support two USB 2.0 ports. By default both ports are configured in USB host mode. This overlay option configure the second port to USB device mode. This option should use with BP_OVERLAY_USB_LED.

This overlay option applies to all chips.

BP_OVERLAY_SPI_SSBn_EXT_CS

BCM6362 Parameters:

- *BP_OVERLAY_SPI_SSB2_EXT_CS*: Enable SPI chip slave select LS_SPI_SSB_2 on GPIO pin 9.
- *BP_OVERLAY_SPI_SSB3_EXT_CS*: Enable SPI chip slave select LS_SPI_SSB_3 on GPIO pin 10.

BCM63268 Parameters:

- *BP_OVERLAY_HS_SPI_SSB4_EXT_CS*: Enable SPI slave chip select SPI_SSB_4 on GPIO pin 16.
- *BP_OVERLAY_HS_SPI_SSB5_EXT_CS*: Enable SPI chip slave select SPI_SSB_5 on GPIO pin 17.
- *BP_OVERLAY_HS_SPI_SSB6_EXT_CS*: Enable SPI chip slave select SPI_SSB_4 on GPIO pin 8.
- *BP_OVERLAY_HS_SPI_SSB7_EXT_CS*: Enable SPI chip slave select SPI_SSB_5 on GPIO pin 9.

BP_OVERLAY_SPI_EXT_CS

These overlay options apply to BCM6328.

For the BCM6328, this overlay option enables the SPI chip slave select function on the SPI_SS_B3 pin. By default this pin is configured as GPIO function.

BP_OVERLAY_PCIE_CLKREQ

This overlay option applies to BCM6328, BCM63268, and BCM6828. It enables the PCIe Clock Request signal PCIE_CLKREQ on GPIO pin. See Table 5 for the detail of the pin assignments.

Table 5: BP_OVERLAY_PCIE_CLKREQ GPIO Pin Assignments

Chip	*GPIO Pin*
BCM6328	GPIO 16
BCM63268	GPIO 23

BP_OVERLAY_UART1

This is option is not used.

bp_usSerialLedData/Clk/Mask

bp_usSerialLedData

bp_usSerialLedClk

bp_usSerialLedMask

Scope

BCM63138, BCM63148, BCM63381

Description

These three APIs replace the function of enabling BP_OVERLAY_SERIAL_LEDS in the bp_ulGpioOverlay parameter. These three signals identify the pins assigned to the clock, data, and (optional) mask that are sent to an external serial shift register for Leds.

The bp_usSerialLedData pin can be defined to be active high or active low indicating that the entire shift register should be inverted.

Table 6: Data, Clock, and Mask GPIO Pins per Chip

	Data	*Clock*	*Mask*
BCM63138	GPIO 0 or 29	GPIO 1 or 30	GPIO 2 or 31
BCM63148	GPIO 0 or 29	GPIO 1 or 30	GPIO 2 or 31
BCM63381	GPIO 17	GPIO 16	GPIO 24

bp_usGpioFpgaReset

Scope

Special test boards only

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for FGPA reset pin

bp_usGpioWirelessPowerDown

Scope

Any board that support wireless module power down

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for wireless module power down

bp_usGpioPassDyingGasp

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the output pin that passes the dying gasp signal to other module on the board.

bp_usGpioUart2Sdin

Scope

BCM63268, BCM63381, BCM63138, BCM63148, BCM6838

Type

Unsigned short

Description

This parameter ID defines the input pin assignment for UART 2 as a standard input. The GPIO number is chip dependent and shown in Table 7.

Table 7: bp_usGpioUart2Sdin GPIO Pin Assignments

Chip	*GPIO Pin*
BCM63268	GPIO_12 or GPIO_26 (active high)
BCM63381	GPIO_04 or GPIO_23 (active high)
BCM63138	GPIO_05 or GPIO_22 (active high)
BCM63148	GPIO_05 or GPIO_22 (active high)
BCM6838X	GPIO_14 (active high)

bp_usGpioUart2Sdout

Scope

BCM63268, BCM63381, BCM63138, BCM63148, BCM6838

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for UART 2 as standard output. The GPIO number is chip dependent and shown in Table 8.

Table 8: bp_usGpioUart2Sdout GPIO Pin Assignments

Chip	*GPIO Pin*
BCM63268	GPIO_13 or GPIO_27 (active high)
BCM63381	GPIO_05 or GPIO_22 (active high)
BCM63138	GPIO_06 or GPIO_23 (active high)
BCM63148	GPIO_06 or GPIO_23 (active high)
BCM6838X	GPIO_15 (active high)

bp_usGpioFemtoReset

Scope

All boards with Femto Chip

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for FEMTO chip reset signal.

Ethernet Switch Parameter ID Definitions

The following parameter IDs define the parameters for internal and external switch configuration on the board. Each switch must define the following parameter IDs:

- *bp_ucPhyType0/1*
- *bp_ucPhyAddress*
- *bp_usConfigType, bp_ulPortMap*
- *bp_ulPhyId*

The *bp_usSpeedLed100* and *bp_usSpeedLed1000* parameters are only required for the internal switch of the BCM6818.

bp_ucPhyType0

Scope

All chips

Type

Unsigned char

Description

This parameter ID defines the internal port connected to the internal switch. For historical reasons, it is always set to *BP_ENET_EXTERNAL_SWITCH*.

bp_ucPhyType1

Scope

All chips

Type

Unsigned char

Description

This parameter ID defines the external switch type. It is always set to *BP_ENET_EXTERNAL_SWITCH* if present.

bp_ucPhyAddress

Scope

All chips

Type

Unsigned char

Description

This parameter is always set to zero.

bp_usConfigType

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the switch configuration type. For internal switch it is always set to *BP_ENET_CONFIG_MMAP*. For external switch, it can be set to the following value the management mechanism for the external switch:

BP_ENET_CONFIG_MDIO

Use MDIO interface to configure external switch

BP_ENET_CONFIG_GPIO_MDIO

Use GPIO simulated MDIO interface to configure external switch

BP_ENET_CONFIG_MDIO_PSEUDO_PHY

Use MDIO PSEDUO PHY to configure external switch

BP_ENET_CONFIG_SPI_SSB_x

Use SPI interface to configure external switch.

- BP_ENET_CONFIG_SPI_SSB_0
- BP_ENET_CONFIG_SPI_SSB_1
- BP_ENET_CONFIG_SPI_SSB_2
- BP_ENET_CONFIG_SPI_SSB_3

Use SPI interface to configure external switch

BP_ENET_CONFIG_HS_SPI_SSB_x

Use High Speed SPI interface to configure external switch:

- BP_ENET_CONFIG_HS_SPI_SSB_0
- BP_ENET_CONFIG_HS_SPI_SSB_1
- BP_ENET_CONFIG_HS_SPI_SSB_2
- BP_ENET_CONFIG_HS_SPI_SSB_3
- BP_ENET_CONFIG_HS_SPI_SSB_4
- BP_ENET_CONFIG_HS_SPI_SSB_5
- BP_ENET_CONFIG_HS_SPI_SSB_6
- BP_ENET_CONFIG_HS_SPI_SSB_7

bp_ulPortMap

Scope

All chips

Type

Unsigned long

Description

This parameter ID defines the bitmap of the ports used in the switch. When a port is populated with RJ45 connector, either a port with integrated PHY or a port using external PHY, or connected to external switch, or connected to WAN interface, the corresponding bit in *bp_ulPortMap* should be set to one. For example, if port 0 and 1 are populated and port 4 is connected to external switch, the *bp_ulPortMap* for the internal switch should be set to 0x13.

Table 9 through Table 11 show the internal switch's port type, mapping, PHY ID, and other properties.

Table 9: BCM63268 Switch Configuration

Port	Name	Type	PHY ID	LED Overlay Flag
0	EPHY1	GEMAC + EPHY	1+EPHY Base Addr	BP_OVERLAY_EPHY_LED_0
1	EPHY2	GEMAC + EPHY	2+EPHY Base Addr	BP_OVERLAY_EPHY_LED_1
2	EPHY3	GEMAC + EPHY	3+EPHY Base Addr	BP_OVERLAY_EPHY_LED_2
3	GPHY1	GEMAC + GPHY	4	BP_OVERLAY_GPHY_LED_0
4	RGMII_1	RGMII/MII/RvMII GEMAC	External	N/A
5	RGMII_2	RGMII GEMAC	External	N/A
6	RGMII_3	RGMII/MII/RvMII GEMAC	External	N/A
7	RGMII_4	RGMII GEMAC	External	N/A

Table 10: BCM6328 Switch Configuration

Port	Name	Type	PHY ID	LED Overlay Flag
0	EPHY1	FEMAC + EPHY	1	BP_OVERLAY_EPHY_LED_0
1	EPHY2	FEMAC + EPHY	2	BP_OVERLAY_EPHY_LED_1
2	EPHY3	FEMAC + EPHY	3	BP_OVERLAY_EPHY_LED_2
3	EPHY4	FEMAC + EPHY	4	BP_OVERLAY_EPHY_LED_3
4	GMII_1	RGMII/MII/RvMII GEMAC	External	N/A

Table 11: BCM6362 Switch Configuration

Port	Name	Type	PHY ID	LED Overlay Flag
0	EPHY1	FEMAC + EPHY	1	BP_OVERLAY_EPHY_LED_0
1	EPHY2	FEMAC + EPHY	2	BP_OVERLAY_EPHY_LED_1
2	EPHY3	FEMAC + EPHY	3	BP_OVERLAY_EPHY_LED_2
3	EPHY4	FEMAC + EPHY	4	BP_OVERLAY_EPHY_LED_3

Table 11: BCM6362 Switch Configuration

Port	Name	Type	PHY ID	LED Overlay Flag
4	GMII_1	RGMII/MII/RvMII GEMAC	External	N/A
5	GMII_2	RGMII GEMAC	External	N/A

Table 12: BCM63138 Switch Configuration

Port	Name	Type	PHY ID
0	GPHY0	GEMAC + GPHY	GPHY_BASE_ADDRESS+0
1	GPHY1	GEMAC + GPHY	GPHY_BASE_ADDRESS+1
1	GPHY2	GEMAC + GPHY	GPHY_BASE_ADDRESS+2
1	GPHY3	GEMAC + GPHY	GPHY_BASE_ADDRESS+3

Each can have bp_usLinkLed, bp_usSpeedLed100, or bp_usSpeedLed1000 parameters.

bp_ulPhyId*n*

Scope

All chips

Type

Unsigned long

Description

These parameter IDs define the PHY ID for corresponding port. Only the eight LSBs are used to represent the actual PHY ID. The upper bits are used for special flags. Refer to *boardparam.h* for all the flag definitions. Specify *bp_ulPhyIdx* in the board parameter set when the port is used.

For internal switch ports with integrated PHY, the port PHY ID is fixed for each port. See the switch configurations in Table 9 through Table 11 for integrated PHY ID on each chip. The PHY ID is defined as:

```
BP_PHY_ID_x | PHY_INTERNAL | PHY_INTEGRATED_VALID
```

where BP_PHY_ID_x is the internal PHY ID, as shown below.

```
{bp_ulPhyId2,  .u.ul = BP_PHY_ID_3 | PHY_INTERNAL | PHY_INTEGRATED_VALID }
```

The PHY ID for the port on the internal switch with external PHY is defined as:

```
BP_PHY_ID_y | MAC_IFACE_VALID | MAC_IF_type | PHY_EXTERNAL| PHY_INTEGRATED_VALID
```

where *BP_PHY_ID_y* is external PHY ID and *MAC_IF_type* can be MAC_IF_RGMII, MAC_IF_GMII, MAC_IF_MII, or MAC_IF_RvMII, as in this example:

```
{bp_ulPhyId4,       .u.ul = BP_PHY_ID_24 | MAC_IFACE_VALID | MAC_IF_RGMII | PHY_INTEGRATED_VALID
| PHY_EXTERNAL},
```

The PHY ID for the port on the external switch is defined same as internal port with integrated PHY.

The software has some backward-compatibility logic that permits existing less explicit definitions to work. For example, the user can just define the PHY ID without any flags if the PHY ID is less than 0x10 for the port with integrated PHY. But new boards should explicitly define the port PHY ID based on rule set forth above.

If an internal switch port is connected to an external switch, the PHY ID should be defined as: Interface_Type|EXTSW_CONNECTED for BCM63268 and BCM6828 chips and Interface_Type for all other chips, where Interface_Type can be RGMII_DIRECT, GMII_DIRECT and MII_DIRECT.

Example:

```
{bp_ulPhyId4,              .u.ul = RGMII_DIRECT | EXTSW_CONNECTED},
```

Note that for the non BCM63268 and BCM6828 chips, the internal switch ports are not available when external switch is used.

On the EPON chip (BCM6828), the user must also define the WAN port 7 and set bit 7 in bp_ulPortMap

.

```
{bp_ulPhyId7,  .u.ul = GMII_DIRECT  | CONNECTED_TO_EPON_MAC}
```

ON the GPON chip (BCM6816), the user also needs to define the WAN port 4 and set bit 4 in bp_ulPortMap.

.

```
{bp_ulPhyId4,                .u.ul = 0xff | PHYCFG_VALID}
```

When connects to other external MII entity such as Femto cell, specify the MII interface type and set the bit for that port in the bp_ulPortMap.

.

```
{bp_ulPortMap,               .u.ul = 0x1f},
{bp_ulPhyId4,                .u.ul = MII_DIRECT},
```

bp_usEphyBaseAddress

Scope

BCM63268

Type

Unsigned short

Description

These parameter IDs define the integrated EPHY base address. The EHPY base address is a 5-bit number. This parameter allows the user to specify the two MSB bits of the base address, so the three LSB bits must be zero.

bp_usGphyBaseAddress

Scope

BCM63138, BCM63148, BCM63268

Type

Unsigned short

Description

These parameter IDs define the integrated GPHY base address. The GHPY base address is a 5-bit number. This parameter allows user to specify the two MSB bits of the base address, so the three LSB bits must be zero.

bp_usDuplexLed

Scope

BCM6818

Type

Unsigned short

Description

This parameter ID is not used.

bp_usSpeedLed100/bp_usSpeedLed1000

Scope

BCM6818

Type

Unsigned short

Description

These parameter IDs define output pin assignment for the integrated GPHY's link speed LEDs.

bp_usPhyConnType

Scope

All

Type

Unsigned short

Description

This parameter is used to identify the phy connection type for a given PHY ID. This field is required for MoCA and PLC. The following values can be used:

- PHY_CONN_TYPE_INT_PHY
- PHY_CONN_TYPE_EXT_PHY
- PHY_CONN_TYPE_EXT_SW
- PHY_CONN_TYPE_EPON
- PHY_CONN_TYPE_GPON
- PHY_CONN_TYPE_MOCA
- PHY_CONN_TYPE_PLC
- PHY_CONN_TYPE_FEMTO
- PHY_CONN_TYPE_MOCA_ETH

bp_usPhyDevName

Scope

All

Type

Unsigned char

Description

This parameter is used to specify an alternative name for a given device. The default name for a port is ETHX. This parameter is required for MoCA and PLC interfaces.

bp_ulPortMaxRate

Scope

BCM6362 and BCM63268

Type

long

Description

This parameter is used to specify a maximum bit rate for a given port. It is intended for use with RGMII connections where the connected device does not support the full rate.

DSL AFE Parameter ID Definitions

bp_ulAfeId0

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned Long

Description

This parameter ID defines internal AFE ID for the board. Refer to *Configuring AFE_ID for xDSL PHY Firmware* (see Reference [3] on page 8) for more details.

bp_ulAfeId1

Scope

BCM6368, BCM63268

Type

Unsigned short

Description

This parameter ID defines external AFE ID for the board. Refer to *Configuring AFE_ID for xDSL PHY Firmware* (see Reference [3] on page 8) for more details.

bp_usGpioExtAFEReset

Scope

BCM6368, BCM63268

Type

Unsigned short

Description

This parameter ID defines output pin assignment for the external AFE reset signal.

bp_usGpioExtAFELDPwr, bp_usGpioExtAFELDMode, bp_usGpioIntAFELDPwr, and bp_usGpioIntAFELDMode

Scope

BCM63268

These parameters must be used in conjunction with the BP_OVERLAY_PHY overlay option.

Type

Unsigned short

Description

These parameter IDs define output pin assignment for the external and internal AFE line driver power and mode control signals. The DSL PHY hardware block has the choices to enable i_gpio_vdsl_ctrl[1:0] and i_gpio_vdsl_ctrl[3:2] output to certain pairs of GPIO pin for line driver control based on these parameters' setting. The following table shows all the possible combinations of the GPIO pin usage:

Table 13: LD Control Pin Combinations

LD Control	*GPIO Pin Assignment*			
vdsl_ctrl[1:0]	GPIO[11:10]	GPIO[11:10]	GPIO[25:24]	GPIO[25:24]
vdsl_ctrl[3:2]	GPIO[13:12]	GPIO[27:26]	GPIO[27:26]	GPIO[13:12]

The user can assign the internal and/or external pair of LD Power and Mode parameter to any pair of GPIO pin pair: [BP_GPIO_10_AH, BP_GPIO_11_AH], [BP_GPIO_12_AH, BP_GPIO_13_AH], [BP_GPIO_24_AH, BP_GPIO_25_AH], [BP_GPIO_26_AH, BP_GPIO_27_AH] based on the values in Table 13. GPIO pins must be assigned in a pair to the line driver mode and power control. If one GPIO pin in the pair is used for one line driver control function, the other GPIO pin in the pair must be used for the other function of line driver control and cannot be used for any other GPIO purpose.

bp_usGpioAFELDRelay

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned short

Description

This parameter ID defines output pin assignment for the AFE LD Relay signal.

bp_usGpioExtAFELDClk

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned short

Description

This parameter ID defines output pin assignment for the external AFE LD clock signal.

bp_usGpioExtAFELDData

Scope

BCM6328, BCM6368, BCM6362, BCM63268

Type

Unsigned short

Description

This parameter ID defines output pin assignment for the external AFE LD data signal.

GPON/EPON Parameter ID Definitions

bp_usGponOpticsType

Scope

BCM6816, BCM6818

Type

Unsigned short

Description

This parameter ID defines GPON optics type. It supports the following types:

- *BP_GPON_OPTICS_TYPE_LEGACY*
- *BP_GPON_OPTICS_TYPE_BOSA*

bp_cpDefaultOpticalParams

Scope

Any board with a Mindspeed M02098 Laser Driver chip

Type

Array of char

Description

This parameter ID defines default optical parameter.

External Interrupt Parameter ID Definitions

bp_usGpio_Intr

Scope

All chips

Type

Unsigned short

Description

This parameter may be used following any interrupt parameter ID, including bp_usButtonExtIntr, bp_usExtIntrSesBtnWireless, bp_usExtIntrResetToDefault, bp_usExtIntrMocaHostIntr, bp_usExtIntrMocaSBIntr0, bp_usExtIntrMocaSBIntr1, bp_usExtIntrLTE, and bp_usExtIntrTrplxrTxFail. If specified, the system will map the previously specified interrupt against a state change on the specified GPIO. This parameter takes values in the range BP_GPIO_0_AL to BP_GPIO_141_AL and GPIO_0_AH to GPIO_141_AH. The AL suffix implies active-low, and the AH suffix implies active-high. The interrupt will be mapped to when the GPIO transitions into its active state.

Example:

```
{bp_usButtonExtIntr,         .u.us = BP_EXT_INTR_1},
{bp_usGpio_Intr,             .u.us = BP_GPIO_72_AL},
```

Will cause Ext Intr 1 to trigger when the GPIO signal 72 transitions to the low state.

bp_usExtIntrResetToDefault

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the external interrupt pin signal that connects to the reset factory default button.

Available values for the BCM6368, BCM6816 and BCM6818 are BP_EXT_INTR_0 to BP_EXT_INTR_5.

Available values for the BCM63268, BCM6328, BCM6362 and BCM6828 are BP_EXT_INTR_0 to BP_EXT_INTR_3.

This parameter may be immediately followed by a bp_usGpio_Intr parameter ID. If so, the system will automatically configure the specified GPIO to trigger the interrupt specified in this parameter.

Note: While still supported, use of the new button parameter IDs is recommended. Refer to the HOWTO/ButtonConfiguration.pdf document included in the release tarball for more details.

bp_usExtIntrSesBtnWireless

Scope

All boards with Wi-Fi support.

Type

Unsigned short

Description

This parameter ID defines the external interrupt pin signal that connects to the reset Wi-Fi protected setup push button.

Available values for the BCM6368, BCM6816 and BCM6818 are BP_EXT_INTR_0 to BP_EXT_INTR_5.

Available values for the BCM63268, BCM6328, BCM6362 and BCM6828 are BP_EXT_INTR_0 to BP_EXT_INTR_3.

This parameter should be immediately followed by a bp_usGpio_Intr parameter ID. If so, the system will automatically configure the specified GPIO to trigger the interrupt specified in this parameter.

Note: While still supported, use of the new button parameter IDs is recommended. Refer to the HOWTO/ButtonConfiguration.pdf document included in the release tarball for more details.

bp_usButtonIdx

Scope

All chips

Type

Unsigned short

Description

This parameter specifies a new button. Its value will be a 0-based index to the button number. This should be less than 3. This parameter id must be followed by bp_usButtonExtIntr and bp_usGpioIntr parameters, to indicate which GPIO and interrupt the button is associated with. Following these, may be one or more bp_usButtonAction and bp_ulButtonActionParm parameter IDs, which associate actions with button events.

Example:

```
  {bp_usButtonIdx,              .u.us = 1},
  {  bp_usButtonExtIntr,        .u.us = BP_EXT_INTR_2 | BP_EXT_INTR_TYPE_IRQ_HIGH_LEVEL },
  {  bp_usGpio_Intr,            .u.us = BP_GPIO_11_AH },
  {    bp_usButtonAction,         .u.us = BP_BTN_ACTION_PRINT | BP_BTN_TRIG_PRESS },
  {      bp_ulButtonActionParm,     .u.ul = (unsigned long)"Hold for 5s to restore to default" },
  {    bp_usButtonAction,         .u.us = BP_BTN_ACTION_RESTORE_DEFAULTS | BP_BTN_TRIG_HOLD |
BP_BTN_TRIG_5S },
```

This creates a button associated with external interrupt 2, on GPIO 11, which is considered pressed when the GPIO signal 11 is high. This further associates two actions with the button. When the button is first pressed, it will trigger the 'print' action, printing the string, and when held for five seconds it will trigger the restore to default action.

In addition, other actions may be associated with the specified button index at runtime using the registerPushButtonPressNotifyHook, registerPushButtonHoldNotifyHook, and registerPushButtonReleaseNotifyHook APIs.

bp_usButtonExtIntr

Scope

All chips

Type

Unsigned short

Description

This parameter must be used following bp_usButtonIdx. It specifies which interrupt the given button is mapped to. This parameter may be immediately followed by a bp_usGpioIntr parameter.

bp_usButtonAction

Scope

All chips

Type

Unsigned short

Description

This parameter registers a button action against a button. It must be specified after a bp_usButtonIdx parameter. It may optionally be followed by a bp_ulButtonActionParm parameter, which will specify a parameter to be passed to the action handler.

The value is made up of two parts, an action and a trigger. The action may be one of the parameters listed in Table 14.

Table 14: Button Action

Parameter	*Description*
BP_BTN_ACTION_NONE	Does nothing. This may be used to invalidate a previous release action after a period of time. If you registered a release action, with time 0, and then registered a NONE action with time 3, then the original release action would only get invoked if the button was held for less than three seconds.
BP_BTN_ACTION_SES	Initiates a wireless SES key exchange. If 1905 is compiled in, this will initiate key exchanges on all 1905 interfaces instead.
BP_BTN_ACTION_PLC_UKE	Initiates a PLC key exchange.
BP_BTN_ACTION_RANDOMIZE_PLC	Causes the PLC to select a new random key.
BP_BTN_ACTION_RESTORE_DEFAULTS	Restores the device to factory default settings, and resets the board.
BP_BTN_ACTION_RESET	Causes the board to reset.
BP_BTN_ACTION_PRINT	Causes a message to be printed to the CLI. Takes a parameter, which will be a pointer to a string (cast to unsigned long).

The trigger may be one of the parameters listed in Table 15.

Table 15: Button Trigger

Parameter	*Description*
BP_BTN_TRIG_PRESS	The action is invoked when the button is first pressed. If multiple actions are registered against this trigger, all actions will occur.
BP_BTN_TRIG_HOLD	The action is invoked when the button is held for a period of time. A button release is not required to cause the event to occur.
BP_BTN_TRIG_RELEASE	The action is invoked after the button is released after a specified time (if no time is specified 0 seconds is assumed). Only the action(s) with the largest timeout less than the release time are invoked. If multiple actions are registered with the same timeout all actions with that timeout will be invoked.

Note: The trigger may be or'ed with a timeout trigger of BP_BTN_TRIG_0s to BP_BTN_TRIG_10s.

bp_usCfeResetToDefaultBtnIdx

Scope

All chips

Type

Unsigned short

Description

This parameter specifies the button index to be used to perform a reset to default when the board is in CFE mode. This is used in conjunction with the bp_usButtonIdx parameter, however, it is not considered a button action; it should not be placed between the bp_usButtonIdx parameter and any bp_usButtonAction parameters which apply to that button. It is not possible to specify a custom trigger for the event – the restore to default will take place immediately when the button is pressed, when the modem is in CFE mode, or if held down over the course of a power cycle.

Voice Parameter ID Definitions

bp_ucDspType0 and bp_ucDspType1

Scope

All chips

Type

Unsigned char

Description

These parameters id define the type of first and second DSP if available. Available values are BP_VOIP_MIPS, BP_VOIP_DSP, BP_VOIP_NO_DSP.

bp_ucDspAddress

Scope

All chips

Type

Unsigned char

Description

These parameters id is always zero.

SPI Slave Parameter ID Definitions

bp_usGpioSpiSlaveReset

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines the output pin assignment for SPI slave reset pin.

bp_usGpioSpiSlaveBootMode

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines an output pin assignment used to control the boot mode of the SPI slave device.

bp_usSpiSlaveBusNum

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI slave bus number. Set to LEG_SPI_BUS_NUM for low-speed SPI slave and HS_SPI_BUS_NUM for high-speed SPI slave.

bp_usSpiSlaveSelectNum

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI slave chip select number. If the specified chip select signal is multiplexed with a GPIO pin, the user must set one of the following overlay options in *bp_usGpioOverlay* depending on target chip:

- BP_OVERLAY_SPI_SSBx_EXT_CS
- BP_OVERLAY_HS_SPI_SSBx_EXT_CS
- BP_OVERLAY_SPI_EXT_CS

See "bp_ulGpioOverlay" on page 26 for more details.

bp_usSpiSlaveSelectGpioNum

Scope

BCM63138, BCM63148, BCM63381

Type

Unsigned short

Description

In BCM63138, BCM63148, BCM63381 chips, there may be multiple choices of the GPIO pin selection for a particular SPI slave selection signal through pinmux setting. This parameter allow user to select which GPIO pin to be used. The board parameter framework automatically set the pinmux properly based on the board parameter configuration.

Also for these chips, it is required to explicitly specify the bp_usSpiSlaveSelectNum and bp_usSpiSlaveSelectGpioNum setting in the board parameter for the SPI interface usage on the voice daughter card.

bp_usSpiSlaveMode

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI slave mode. It can be one of the following values:

- SPI_MODE_0 (0)
- SPI_MODE_1 (SPI_CPHA)
- SPI_MODE_2 (SPI_CPOL)
- SPI_MODE_3 (SPI_CPOL|SPI_CPHA).

bp_ulSpiSlaveCtrlState

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI controller state. It can be one of the following values:

- SPI_CONTROLLER_STATE_SET
- SPI_CONTROLLER_STATE_CPHA_EXT
- SPI_CONTROLLER_STATE_GATE_CLK_SSOFF
- SPI_CONTROLLER_STATE_ASYNC_CLOCK

bp_ulSpiSlaveMaxFreq

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI slave maximum clock rate in Hertz (Hz).

bp_usSpiSlaveProtoRev

Scope

All chips

Type

Unsigned short

Description

This parameter ID defines SPI slave protocol revision.

Wireless Parameter ID Definitions

bp_usAntInUseWireless

Scope

All boards with wireless support.

Type

Unsigned short

Description

This parameter ID defines wireless antenna setting. The following values are supported:

- BP_WLAN_ANT_MAIN
- BP_WLAN_ANT_AUX
- BP_WLAN_ANT_BOTH

bp_usWirelessFlags

Scope

All boards with wireless support.

Type

Unsigned short

Description

This parameter ID defines wireless flag. The following values are supported:

- BP_WLAN_MAC_ADDR_OVERRIDE
- BP_WLAN_EXCLUDE_ONBOARD
- BP_WLAN_EXCLUDE_ONBOARD_FORCE
- BP_WLAN_USE_OTP

Miscellaneous Parameter ID Definitions

bp_usVregSel1P2

Scope

Femtocell test board

Type

Unsigned short

Description

This parameter ID is not supported.

bp_ulInterfaceEnable

Scope

BCM63138, BCM63148, BCM63381

Type

Unsigned long

Description

This parameter configures the hardware to explicitly enable a particular interface that would not otherwise be enabled. For example, by setting this to BP_PINMUX_FNTYPE_NAND, the pins used for NAND will be connected to the NAND controller even if the device was strapped to boot from SPI.

Available interface enables:

BP_PINMUX_FNTYPE_HS_SPI | chip_select_number

BP_PINMUX_FNTYPE_IRQ | irq_number

BP_PINMUX_FNTYPE_NAND

BP_PINMUX_FNTYPE_SATA

BP_PINMUX_FNTYPE_DECT

bp_usSgmiiDetect

Scope

BCM963148

Type

Unsigned short

Description

Two values BP_GPIO_28_AH or BP_GPIO_36_AH can be assigned to this variable. This parameter will indicate GPIO 26 or GPIO 28 is used for SerDes Fiber Signal Detection. The value needs to be configured correctly based on the board design.

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

Phone: 949-926-5000
Fax: 949-926-5203
E-mail: info@broadcom.com
Web: www.broadcom.com

963XX-SWUM402-R July 28, 2014